Finding Linear Structure in Large Datasets with Scalable Canonical Correlation Analysis

Zhuang Ma ZHUANGMA@WHARTON.UPENN.EDU
Yichao Lu YICHAOLU@WHARTON.UPENN.EDU
Dean Foster DEAN@FOSTER.NET
Department of Statistics, The Wharton School, University of Pennsylvania, Philadelphia, PA 19104 U.S.A

Abstract

Canonical Correlation Analysis (CCA) is a widely used spectral technique for finding correlation structures in multi-view datasets. In this paper, we tackle the problem of large scale CCA, where classical algorithms, usually requiring computing the product of two huge matrices and huge matrix decomposition, are computationally and storage expensive. We recast CCA from a novel perspective and propose a scalable and memory efficient *Augmented Approximate Gradient (AppGrad)* scheme for finding top k dimensional canonical subspace which only involves large matrix multiplying a thin matrix of width k and small matrix decomposition of dimension $k \times k$. Further, *AppGrad* achieves optimal storage complexity $O(k(p_1+p_2))$, compared with classical algorithms which usually require $O(p_1^2 + p_2^2)$ space to store two dense whitening matrices. The proposed scheme naturally generalizes to stochastic optimization regime, especially efficient for huge datasets where batch algorithms are prohibitive. The online property of stochastic *AppGrad* is also well suited to the streaming scenario, where data comes sequentially. To the best of our knowledge, it is the first stochastic algorithm for CCA. Experiments on four real data sets are provided to show the effectiveness of the proposed methods.

1. Introduction

1.1. Background

Canonical Correlation Analysis (CCA), first introduced in 1936 by (Hotelling, 1936), is a foundamental statistical tool to characterize the relationship between two multidimensional variables, which finds a wide range of applications. For example, CCA naturally fits into multi-view learning tasks and tailored to generate low dimensional feature representations using abandunt and inexpensive unlabeled datasets to supplement or refine the expensive labeled data in a semi-supervised fashion. Improved generalization accuracy has been witnessed or proved in areas such as regression (Kakade & Foster, 2007), clustering (Chaudhuri et al., 2009; Blaschko & Lampert, 2008), dimension reduction (Foster et al., 2008; McWilliams et al., 2013), word embeddings (Dhillon et al., 2011; 2012), etc. Besides, CCA has also been succesfully applied to genome-wide association study (GWAS) and has been shown powerful for understanding the relationship between genetic variations and phenotypes (Witten et al., 2009; Chen et al., 2012).

There are various equivalent ways to define CCA and here we use the linear algebraic formulation of (Golub & Zha, 1995), which captures the very essense of the procedure, pursuing the directions of maximal correlations between two data matrices.

Definition 1.1. *For data matrices* $\mathbf{X} \in \mathbb{R}^{n\times p_1}, \mathbf{Y} \in \mathbb{R}^{n\times p_2}$ *Let* $\mathbf{S_x} = \mathbf{X}^\top\mathbf{X}/n$, $\mathbf{S_y} = \mathbf{Y}^\top\mathbf{Y}/n$, $\mathbf{S_{xy}} = \mathbf{X}^\top\mathbf{Y}/n$ *and* $p = \min\{p_1, p_2\}$. *The canonical correlations* $\lambda_1, \cdots, \lambda_p$ *and corresponding pair of canonical vectors* $\{(\phi_i, \psi_i)\}_{i=1}^p$ *between* $\mathbf{X}$ *and* $\mathbf{Y}$ *are defined recursively by*

$$(\phi_j, \psi_j) = \arg \max_{\substack{\phi^\top\mathbf{S_x}\phi=1,\, \psi^\top\mathbf{S_y}\psi=1 \\ \phi^\top\mathbf{S_x}\phi_i=0,\, \psi^\top\mathbf{S_y}\psi_i=0,\, 1\le i\le j-1}} \phi^\top\mathbf{S_{xy}}\psi$$

$$\lambda_j = \phi_j^\top\mathbf{S_{xy}}\psi_j \quad j = 1, \cdots, p$$

Lemma 1.1. *Let* $\mathbf{S_x}^{-\frac{1}{2}}\mathbf{S_{xy}}\mathbf{S_y}^{-\frac{1}{2}} = \mathbf{UDV}^\top$ *be the singular value decomposition. Then* $\mathbf{\Phi} = \mathbf{S_x}^{-\frac{1}{2}}\mathbf{U}$, $\mathbf{\Psi} = \mathbf{S_y}^{-\frac{1}{2}}\mathbf{V}$, *and* $\mathbf{\Lambda} = \mathbf{D}$ *where* $\mathbf{\Phi} = (\phi_1, \cdots, \phi_p)$, $\mathbf{\Psi} = (\psi_1, \cdots, \psi_p)$ *and* $\mathbf{\Lambda} = diag(\lambda_1, \cdots, \lambda_p)$.

The identifiability of canonical vectors $(\mathbf{\Phi}, \mathbf{\Psi})$ is equivalent to the identifiability of the singular vectors $(\mathbf{U}, \mathbf{V})$.

Proceedings of the 32^{nd} International Conference on Machine Learning, Lille, France, 2015. JMLR: W&CP volume 37.

Lemma 1.1 implies that the leading k dimensional CCA subspace can be solved by first computing the whitening matrices $\mathbf{S_x}^{-\frac{1}{2}}, \mathbf{S_y}^{-\frac{1}{2}}$ and then perform a k-truncated SVD on the whitened covariance matrix $\mathbf{S_x}^{-\frac{1}{2}}\mathbf{S_{xy}}\mathbf{S_y}^{-\frac{1}{2}}$. This classical algorithm is feasible and accurate when the data matrices are small but it can be slow and numerically unstable for large scale datasets which are common in modern natural language processing (large corpora, Dhillon et al. (2011; 2012)) and multi-view learning (abandunt and inexpensive unlabeled data, Hariharan & Subramanian (2014)) applications.

Throughout the paper, we call the step of orthonormalizing the columns of $\mathbf{X}$ and $\mathbf{Y}$ **whitening step**. The computational complexity of the classical algorithm is dominated by the whitening step. There are two major bottlenecks,

- Huge matrix multiplication $\mathbf{X}^\top\mathbf{X}, \mathbf{Y}^\top\mathbf{Y}$ to obtain $\mathbf{S_x}, \mathbf{S_y}$ with computational complexity $O(np_1^2 + np_2^2)$ for general dense $\mathbf{X}$ and $\mathbf{Y}$.

- Large matrix decomposition to compute $\mathbf{S_x}^{-\frac{1}{2}}$ and $\mathbf{S_y}^{-\frac{1}{2}}$ with computational complexity $O(p_1^3 + p_2^3)$ (Even when $\mathbf{X}$ and $\mathbf{Y}$ are sparse, $\mathbf{S_x}, \mathbf{S_y}$ are not necessarily sparse)

Remark 1.1. *The whitening step dominates the k-truncated SVD step because the top k dimensional singular vectors can be efficiently computed by randomized SVD algorithms (see Halko et al. (2011) and many others).*

Remark 1.2. *Another classical algorithm (built-in function in Matlab) introduced in (Björck & Golub, 1973) uses a different way of whitening. It first carrys out a QR decomposition, $\mathbf{X} = \mathbf{Q}_x\mathbf{R}_x$ and $\mathbf{Y} = \mathbf{Q}_y\mathbf{R}_y$ and then performs a SVD on $\mathbf{Q}_x^\top\mathbf{Q}_y$, which has the same computational complexity $O(np_1^2 + np_2^2)$ as the algorithm indicated by Lemma 1.1. However, it is difficult to exploit sparsity in QR factorization while $\mathbf{X}^\top\mathbf{X}, \mathbf{Y}^\top\mathbf{Y}$ can be efficiently computed when $\mathbf{X}$ and $\mathbf{Y}$ are sparse.*

Besides computational issues, extra $O(p_1^2 + p_2^2)$ space is necessary to store two whitening matrices $\mathbf{S_x}^{-\frac{1}{2}}$ and $\mathbf{S_y}^{-\frac{1}{2}}$ (typically dense). In high dimensional applications where the number of features is huge, this can be another bottleneck considering the capacity of RAM of personal desktops (10-20 GB). In large distributed storage systems, the extra required space might incur heavy communication cost.

Therefore, it is natural to ask: is there a scalable algorithm that avoids huge matrix decomposition and huge matrix multiplication? Is it memory efficient? Or even more ambitiously, is there an online algorithm that generates decent approximation given a fixed computational power (e.g. CPU time, FLOP)?

1.2. Related Work

Scalability begins to play an increasingly important role in modern machine learning applications and draws more and more attention. Recently lots of promising progress emerged in the literature concerning with randomized algorithms for large scale matrix approximations, SVD, and Principal Component Analysis (Sarlos, 2006; Liberty et al., 2007; Woolfe et al., 2008; Halko et al., 2011). Unfortunately, these techniques does not directly solve CCA due to the whitening step. Several authors have tried to devise a scalable CCA algorithm. Avron et al. (2013) proposed an efficient approach for CCA between two tall and thin matrices ($p_1, p_2 \ll n$) harnessing the recently developed tools, *Subsampled Randomized Hadamard Transform*, which only subsampled a small proportion of the n data points to approximate the matrix product. However, when the size of the features, p_1 and p_2, are large, the sampling scheme does not work. Later, Lu & Foster (2014) consider sparse design matrices and formulate CCA as iterative least squares, where in each iteration a fast regression algorithm that exploits sparsity is applied.

Another related line of research considers stochastic optimization algorithms for PCA (Arora et al., 2012; Mitliagkas et al., 2013; Balsubramani et al., 2013), which date back to Oja & Karhunen (1985). Compared with batch algorithms, the stochastic versions empirically converge much faster with similar accuracy. Further, these stochastic algorithms can be applied to streaming setting where data comes sequentially (one pass or several pass) without being stored. As mentioned in (Arora et al., 2012), stochastic optimization algorithm for CCA is more challenging and remains an open problem because of the whitening step.

1.3. Main Contribution

The main contribution of this paper is to directly tackle CCA as a nonconvex optimization problem and propose a novel Augmented Approximate Gradient (*AppGrad*) scheme and its stochastic variant for finding the top k dimensional canonical subspace. Its advantages over state-of-art CCA algorithms are three folds. *Firstly*, *AppGrad* scheme only involves large matrix multiplying a thin matrix of width k and small matrix decomposition of dimension $k \times k$, and therefore to some extent is free from the two bottlenecks. It also benefits if $\mathbf{X}$ and $\mathbf{Y}$ are sparse while classical algorithm still needs to invert the dense matrices $\mathbf{X}^\top\mathbf{X}$ and $\mathbf{Y}^\top\mathbf{Y}$. *Secondly*, *AppGrad* achieves optimal storage complexity $O(k(p_1 + p_2))$, the space necessary to store the output, compared with classical algorithms which usually require $O(p_1^2 + p_2^2)$ for storing the whitening matrices. *Thirdly*, the stochastic (online) variant of *AppGrad* is especially efficient for large scale datasets if moderate accuracy is desired. It is well-suited to the case when com-

putational resources are limited or data comes as a stream. To the best of our knowledge, it is the first stochastic algorithm for CCA, which partly gives an affirmative answer to a question left open in (Arora et al., 2012).

The rest of the paper is organized as follows. We introduce *AppGrad* scheme and establish its convergence properties in section 2. We extend the algorithm to stochastic settings in section 3. Extensive real data experiments are presented in section 4. Concluding remarks and future work are summarized in section 5. Proof of Theorem 2.1 and Proposition 2.3 are relegated to the supplementary material.

2. Algorithm

For simplicity, we first focus on the leading canonical pair (ϕ_1, ψ_1) to motivate the proposed algorithms. Results for general scenario can be obtained in the same manner and will be briefly discussed in the later part of this section.

2.1. An Optimization Perspective

Throughout the paper, we assume $\mathbf{X}$ and $\mathbf{Y}$ are of full rank. We use $\|\cdot\|$ for L_2 norm. $\forall\, u \in \mathbb{R}^{p_1}$, $v \in \mathbb{R}^{p_2}$, we define $\|u\|_x = (u^\top \mathbf{S_x} u)^{\frac{1}{2}}$ and $\|v\|_y = (v^\top \mathbf{S_y} v)^{\frac{1}{2}}$, which are norms induced by $\mathbf{X}$ and $\mathbf{Y}$.

To begin with, we recast CCA as an nonconvex optimization problem (Golub & Zha, 1995).

Lemma 2.1. *(ϕ_1, ψ_1) is the solution of*

$$\begin{aligned} &\min \frac{1}{2n}\|\mathbf{X}\phi - \mathbf{Y}\psi\|^2 \\ &\textit{subject to} \quad \phi^\top \mathbf{S_x}\phi = 1,\ \psi^\top \mathbf{S_y}\psi = 1 \end{aligned} \tag{1}$$

Although (1) is a nonconvex (due to the nonconvex constraint), (Golub & Zha, 1995) showed that an alternating minimization strategy (Algorithm 1), or rather iterative least squares, actually converges to the leading canonical pair. However, each update $\phi^{t+1} = \mathbf{S_x^{-1}S_{xy}}\psi^t$ is computationally intensive. Essentially, the alternating least squares acts like a second order method, which is usually recognized to be inefficient for large-scale datasets, especially when current estimate is not close enough to the optimum. Therefore, it is natural to ask: is there a valid first order method that solves (1)? Heuristics borrowed from convex optimization literature give rise to a projected gradient scheme summarized in Algorithm 2. Instead of completely solving a least squares in each iterate, a single gradient step of (1) is performed and then project back to the constrained domain, which avoids inverting a huge matrix. Unfortunately, the following proposition demonstrates that Algorithm 2 fails to converge to the leading canonical pair.

Algorithm 1 CCA via Alternating Least Squares

Input: Data matrix $\mathbf{X} \in \mathcal{R}^{n\times p_1}, \mathbf{Y} \in \mathcal{R}^{n\times p_2}$ and initialization (ϕ^0, ψ^0)
Output : $(\phi_{\text{ALS}}, \psi_{\text{ALS}})$
repeat
$\phi^{t+1} = \arg\min_{\phi} \frac{1}{2n}\|\mathbf{X}\phi - \mathbf{Y}\psi^t\|^2 = \mathbf{S_x^{-1}S_{xy}}\psi^t$
$\phi^{t+1} = \phi^{t+1}/\|\phi^{t+1}\|_x$
$\psi^{t+1} = \arg\min_{\psi} \frac{1}{2n}\|\mathbf{Y}\psi - \mathbf{X}\phi^t\|^2 = \mathbf{S_y^{-1}S_{yx}}\phi^t$
$\psi^{t+1} = \psi^{t+1}/\|\psi^{t+1}\|_y$
until convergence

Algorithm 2 CCA via Naive Gradient Descent

Input: Data matrix $\mathbf{X} \in \mathcal{R}^{n\times p_1}, \mathbf{Y} \in \mathcal{R}^{n\times p_2}$, initialization (ϕ^0, ψ^0), step size η_1, η_2
Output : NAN (incorrect algorithm)
repeat
$\phi^{t+1} = \phi^t - \eta_1 \mathbf{X}^\top(\mathbf{X}\phi^t - \mathbf{Y}\psi^t)/n$
$\phi^{t+1} = \phi^{t+1}/\|\phi^{t+1}\|_x$
$\psi^{t+1} = \psi^t - \eta_2 \mathbf{Y}^\top(\mathbf{Y}\psi^t - \mathbf{X}\phi^t)/n$
$\psi^{t+1} = \psi^{t+1}/\|\psi^{t+1}\|_y$
until convergence

Proposition 2.1. *If leading canonical correlation $\lambda_1 \neq 1$ and either ϕ_1 is not an eigenvector of $\mathbf{S_x}$ or ψ_1 is not an eigenvector of $\mathbf{S_y}$, then $\forall \eta_1, \eta_2 > 0$, the leading canonical pair (ϕ_1, ψ_1) is not a fixed point of the naive gradient scheme in Algorithm 2. Therefore, the algorithm does not converge to (ϕ_1, ψ_1).*

Proof of Proposition 2.1. The proof is similar to the proof of Proposition 2.2 and we leave out the details here. □

The failure of Algorithm 2 is due to the nonconvex nature of (1). Although every gradient step might decrease the objective function, this property no longer persists after projecting to its nonconvex domain $\left\{(\phi, \psi) \,|\, \phi^\top \mathbf{S_x}\phi = 1,\ \psi^\top \mathbf{S_y}\psi = 1\right\}$ (the normalization step). On the contrary, decreases triggered by gradient descent is always maintained if projecting to a convex region.

2.2. *AppGrad* Scheme

As a remedy, we propose a novel Augmented Approximate Gradient (*AppGrad*) scheme summarized in Algorithm 3. It inherits the convergence guarantee of alternating least squares as well as the scalability and memory efficiency of first order methods, which only involves matrix-vector multiplication and only requires $O(p_1 + p_2)$ extra space.

AppGrad seems unnatural at first sight but has some nice intuitions behind as we will discuss later. The differences and similarities between these algorithms are subtle but

Algorithm 3 CCA via AppGrad

Input: Data matrix $\mathbf{X} \in \mathcal{R}^{n\times p_1}, \mathbf{Y} \in \mathcal{R}^{n\times p_2}$, initialization $(\phi^0, \psi^0, \widetilde{\phi}^0, \widetilde{\psi}^0)$, step size η_1, η_2
Output: $(\phi_{\text{AG}}, \psi_{\text{AG}}, \widetilde{\phi}_{\text{AG}}, \widetilde{\psi}_{\text{AG}})$
repeat
$\widetilde{\phi}^{t+1} = \widetilde{\phi}^t - \eta_1 \mathbf{X}^\top(\mathbf{X}\widetilde{\phi}^t - \mathbf{Y}\psi^t)/n$
$\phi^{t+1} = \widetilde{\phi}^{t+1}/\|\widetilde{\phi}^{t+1}\|_x$
$\widetilde{\psi}^{t+1} = \widetilde{\psi}^t - \eta_2 \mathbf{Y}^\top(\mathbf{Y}\widetilde{\psi}^t - \mathbf{X}\phi^t)/n$
$\phi^{t+1} = \widetilde{\psi}^{t+1}/\|\widetilde{\psi}^{t+1}\|_y$
until convergence

crucial. Compared with the naive gradient descent, we introduce two auxiliary variables $(\widetilde{\phi}^t, \widetilde{\psi}^t)$, an unnormalized version of (ϕ^t, ψ^t). During each iterate, we keep updating $\widetilde{\phi}^t$ and $\widetilde{\psi}^t$ without scaling them to have unit norm, which in turn produces the 'correct' normalized counterpart, (ϕ^t, ψ^t). It turns out that $(\phi_1, \psi_1, \lambda_1\phi_1, \lambda_1\psi_1)$ is a fixed point of the dynamic system $\{(\phi^t, \psi^t, \widetilde{\phi}^t, \widetilde{\psi}^t)\}_{t=0}^{\infty}$.

Proposition 2.2. *$\forall\, i \le p$, let $\widetilde{\phi}_i = \lambda_i\phi_i, \widetilde{\psi}_i = \lambda_i\psi_i$, then $(\phi_i, \psi_i, \widetilde{\phi}_i, \widetilde{\psi}_i)$ are the fixed points of AppGrad scheme.*

To prove the proposition, we need the following lemma that characterizes the relations among some key quantities.

Lemma 2.2. $\mathbf{S_{xy}} = \mathbf{S_x}\mathbf{\Phi}\mathbf{\Lambda}\mathbf{\Psi}^\top\mathbf{S_y}$

Proof of Lemma 2.2. By Lemma 1.1, $\mathbf{S_x}^{-\frac{1}{2}}\mathbf{S_{xy}}\mathbf{S_y}^{-\frac{1}{2}} = \mathbf{U}\mathbf{D}\mathbf{V}^\top$, where $\mathbf{U} = \mathbf{S_x}^{\frac{1}{2}}\mathbf{\Phi}$, $\mathbf{V} = \mathbf{S_y}^{\frac{1}{2}}\mathbf{\Psi}$ and $\mathbf{D} = \mathbf{\Lambda}$. Then we have $\mathbf{S_{xy}} = \mathbf{S_x}^{\frac{1}{2}}\mathbf{U}\mathbf{D}\mathbf{V}^\top\mathbf{S_y}^{\frac{1}{2}} = \mathbf{S_x}\mathbf{\Phi}\mathbf{\Lambda}\mathbf{\Psi}^\top\mathbf{S_y}$. □

Proof of Proposition 2.2. Substitute $(\phi^t, \psi^t, \widetilde{\phi}^t, \widetilde{\psi}^t) = (\phi_i, \psi_i, \widetilde{\phi}_i, \widetilde{\psi}_i)$ into the iterative formula in Algorithm 3.

$$\begin{aligned}\widetilde{\phi}^{t+1} &= \widetilde{\phi}_i - \eta_1(\mathbf{S_x}\widetilde{\phi}_i - \mathbf{S_{xy}}\psi_i)\\ &= \widetilde{\phi}_i - \eta_1(\mathbf{S_x}\widetilde{\phi}_i - \mathbf{S_x}\mathbf{\Phi}\mathbf{\Lambda}\mathbf{\Psi}^\top\mathbf{S_y}\psi_i)\\ &= \widetilde{\phi}_i - \eta_1(\mathbf{S_x}\widetilde{\phi}_i - \lambda_i\mathbf{S_x}\phi_i)\\ &= \widetilde{\phi}_i\end{aligned}$$

The second equality is direct application of Lemma 2.2. The third equality is due to the fact that $\mathbf{\Psi}^\top\mathbf{S_y}\mathbf{\Psi} = I_p$. Then,

$$\phi^{t+1} = \widetilde{\phi}_i/\|\widetilde{\phi}_i\|_x = \widetilde{\phi}_i/\lambda_i = \phi_i$$

Therefore $(\widetilde{\phi}^{t+1}, \phi^{t+1}) = (\widetilde{\phi}^t, \phi^t) = (\widetilde{\phi}_i, \phi_i)$. A symmetric argument will show that $(\widetilde{\psi}^{t+1}, \psi^{t+1}) = (\widetilde{\psi}^t, \psi^t) = (\widetilde{\psi}_i, \psi_i)$, which completes the proof. □

The connection between *AppGrad* and alternating minimization strategy is not instaneous. Intuitively, when (ϕ^t, ψ^t) is not close to (ϕ_1, ψ_1), solving the least squares completely as carried out in Algorithm 1 is a waste of computational power (informally by regarding it as a second order method, the Newton Step has fast convergence only when current estimate is close to the optimum). Instead of solving a sequence of possibly unrelevant least squares, the following lemma shows that *AppGrad* directly targets at the least squares that involves the leading canonical pair.

Lemma 2.3. *Let (ϕ_1, ψ_1) be the leading canonical pair and $(\widetilde{\phi}_1, \widetilde{\psi}_1) = \lambda_1(\phi_1, \psi_1)$. Then,*

$$\begin{aligned}\widetilde{\phi}_1 &= \arg\min_{\phi} \frac{1}{2n}\|\mathbf{X}\phi - \mathbf{Y}\psi_1\|^2\\ \widetilde{\psi}_1 &= \arg\min_{\psi} \frac{1}{2n}\|\mathbf{Y}\psi - \mathbf{X}\phi_1\|^2\end{aligned} \tag{2}$$

Proof of Lemma 2.3. Let $\phi^* = \arg\min_{\phi} \frac{1}{2n}\|\mathbf{X}\phi - \mathbf{Y}\psi_1\|^2$, by optimality condition, $\mathbf{S_x}\phi^* = \mathbf{S_{xy}}\psi_1$. Apply Lemma 2.2,

$$\phi^* = \mathbf{S_x}^{-1}\mathbf{S_x}\mathbf{\Phi}\mathbf{\Lambda}\mathbf{\Psi}^\top\mathbf{S_y}\psi_1 = \lambda_1\phi_1 = \widetilde{\phi}_1$$

Similar argument gives $\psi^* = \widetilde{\psi}_1$ □

Lemma 2.3 characterizes the relationship between leading canonical pair (ϕ_1, ψ_1) and its unnormalized counterpart $(\widetilde{\phi}_1, \widetilde{\psi}_1)$, which sheds some insight on how *AppGrad* works. The intuition is that (ϕ^t, ψ^t) and $(\widetilde{\phi}^t, \widetilde{\psi}^t)$ are current estimations of (ϕ_1, ψ_1) and $(\widetilde{\phi}_1, \widetilde{\psi}_1)$, and the updates of $(\widetilde{\phi}^{t+1}, \widetilde{\psi}^{t+1})$ in Algorithm 3 are actually gradient steps of the least squares in (2), with the unknown truth (ϕ_1, ψ_1) approximated by (ϕ^t, ψ^t). In terms of mathematics,

$$\begin{aligned}\widetilde{\phi}^{t+1} &= \widetilde{\phi}^t - \eta_1\mathbf{X}^\top(\mathbf{X}\widetilde{\phi}^t - \mathbf{Y}\psi^t)/n\\ &\approx \widetilde{\phi}^t - \eta_1\mathbf{X}^\top(\mathbf{X}\widetilde{\phi}^t - \mathbf{Y}\psi_1)/n\\ &= \widetilde{\phi}^t - \eta_1\nabla_\phi\frac{1}{2n}\|\mathbf{X}\phi - \mathbf{Y}\psi_1\|^2|_{\phi=\widetilde{\phi}^t}\end{aligned} \tag{3}$$

The normalization step in Algorithm 3 corresponds to generating new approximations of (ϕ_1, ψ_1), namely (ϕ^{t+1}, ψ^{t+1}), using the updated $(\widetilde{\phi}^{t+1}, \widetilde{\psi}^{t+1})$ through the relationship $(\phi_1, \psi_1) = (\widetilde{\phi}_1/\|\widetilde{\phi}_1\|_x, \widetilde{\psi}_1/\|\widetilde{\psi}_1\|_y)$. Therefore, one can interpret *AppGrad* as approximate gradient scheme for solving (2). When $(\widetilde{\phi}^t, \widetilde{\psi}^t)$ converge to $(\widetilde{\phi}_1, \widetilde{\psi}_1)$, its scaled version (ϕ^t, ψ^t) converge to the leading canonical pair (ϕ_1, ψ_1).

The following theorem shows that when the estimates enter a neighborhood of the true canonical pair, *AppGrad* is contractive. Define the error metric $e_t = \|\Delta\widetilde{\phi}^t\|^2 + \|\Delta\widetilde{\psi}^t\|^2$ where $\Delta\widetilde{\phi}^t = \widetilde{\phi}^t - \widetilde{\phi}_1, \Delta\widetilde{\psi}^t = \widetilde{\psi}^t - \widetilde{\psi}_1$.

Theorem 2.1. *Assume $\lambda_1 > \lambda_2$, $\exists\, L_1, L_2 \ge 1$ such that $\lambda_{max}(\mathbf{S_x}), \lambda_{max}(\mathbf{S_y}) \le L_1$ and $\lambda_{min}(\mathbf{S_x}), \lambda_{min}(\mathbf{S_y}) \ge L_2^{-1}$, where $\lambda_{min}(\cdot), \lambda_{max}(\cdot)$ denote smallest and largest eigenvalues. If $e_0 < 2(\lambda_1^2 - \lambda_2^2)/L_1$ and set $\eta_1 = \eta_2 =$*

Algorithm 4 CCA via *AppGrad* (Rank-k)

Input: Data matrix $\mathbf{X} \in \mathcal{R}^{n\times p_1}, \mathbf{Y} \in \mathcal{R}^{n\times p_2}$, initialization $(\mathbf{\Phi}^0, \mathbf{\Psi}^0, \widetilde{\mathbf{\Phi}}^0, \widetilde{\mathbf{\Psi}}^0)$, step size η_1, η_2
Output : $(\mathbf{\Phi}_{\text{AG}}, \mathbf{\Psi}_{\text{AG}}, \widetilde{\mathbf{\Phi}}_{\text{AG}}, \widetilde{\mathbf{\Psi}}_{\text{AG}})$
repeat
$\widetilde{\mathbf{\Phi}}^{t+1} = \widetilde{\mathbf{\Phi}}^t - \eta_1 \mathbf{X}^\top(\mathbf{X}\widetilde{\mathbf{\Phi}}^t - \mathbf{Y}\mathbf{\Psi}^t)/n$
SVD: $(\widetilde{\mathbf{\Phi}}^{t+1})^\top \mathbf{S}_{\mathbf{x}} \widetilde{\mathbf{\Phi}}^{t+1} = \mathbf{U}_x \mathbf{D}_x \mathbf{U}_x^\top$
$\mathbf{\Phi}^{t+1} = \widetilde{\mathbf{\Phi}}^{t+1} \mathbf{U}_x \mathbf{D}_x^{-\frac{1}{2}} \mathbf{U}_x^\top$
$\widetilde{\mathbf{\Psi}}^{t+1} = \widetilde{\mathbf{\Psi}}^t - \eta_2 \mathbf{Y}^\top(\mathbf{Y}\widetilde{\mathbf{\Psi}}^t - \mathbf{X}\mathbf{\Phi}^t)/n$
SVD: $(\widetilde{\mathbf{\Psi}}^{t+1})^\top \mathbf{S}_{\mathbf{y}} \widetilde{\mathbf{\Psi}}^{t+1} = \mathbf{U}_y \mathbf{D}_y \mathbf{U}_y^\top$
$\mathbf{\Psi}^{t+1} = \widetilde{\mathbf{\Psi}}^{t+1} \mathbf{U}_y \mathbf{D}_y^{-\frac{1}{2}} \mathbf{U}_y^\top$
until convergence

$\eta = \delta/6L_1$, then AppGrad achieves linear convergence such that $\forall\, t \in \mathbb{N}_+$

$$e_t \le \left(1 - \frac{\delta^2}{6L_1L_2}\right)^t e_0$$

where $\delta = 1 - \left(1 - \frac{2(\lambda_1^2-\lambda_2^2)-L_1e_0}{2\lambda_1^2}\right)^{\frac{1}{2}} > 0$

Remark 2.1. *The theorem reveals that the larger is the eigengap $\lambda_1 - \lambda_2$, the broader is the contraction region. We didn't try to optimize the conditions above and empirically as shown in the experiments, a randomized initialization always suffices to capture most of the correlations.*

2.3. General Rank-k Case

Following the spirit of rank-one case, *AppGrad* can be easily generalized to compute the top k dimesional canonical subspace as summarized in Algorithm 4. The only difference is that the original scalar normalization is replaced by its matrix counterpart, that is to multiply the inverse of the square root matrix $\mathbf{\Phi}^{t+1} = \widetilde{\mathbf{\Phi}}^{t+1}\mathbf{U}_x\mathbf{D}_x^{-\frac{1}{2}}\mathbf{U}_x^\top$, ensuring that $(\mathbf{\Phi}^{t+1})^\top \mathbf{X}^\top \mathbf{X} \mathbf{\Phi}^{t+1} = \mathbf{I}_k$.

Notice that the gradient step only involves a large matrix multiplying a thin matrix of width k and the SVD is performed on a small $k \times k$ matrix. Therefore, the computational complexity per iteration is dominated by the gradient step, of order $O(n(p_1 + p_2)k)$. The cost will be further reduced when the data matrices $\mathbf{X}, \mathbf{Y}$ are sparse.

Compared with classical spectral agorithm which first whitens the data matrices and then performs a SVD on the whitened covariance matrix, *AppGrad* actually merges these two steps together. This is the key of its efficiency. In a high level, whitening the whole data matrix is not necessary and we only want to whiten the directions that contain the leading CCA subspace. However, these directions are unknown and therefore for two-step procedures, whitening the whole data matrix is unavoidable. Instead, *AppGrad* tries to identify (gradient step) and whiten (normalization step) these directions simultaneously. In this way, every normalization step is only performed on the potential k dimensional target CCA subspace and therefore only deals with a small $k \times k$ matrix.

Parallel results of Lemma 2.1, Proposition 2.1, Proposition 2.2, Lemma 2.3 for this general scenario can be established in a similar manner. Here, to make Algorithm 4 more clear, we state the fixed point result of which the proof is similar to Proposition 2.2.

Proposition 2.3. *Let $\mathbf{\Lambda}_k = diag(\lambda_1, \cdots, \lambda_k)$ be the diagonal matrix of top k canonical correlations and let $\mathbf{\Phi}_k = (\phi_1, \cdots, \phi_k), \mathbf{\Psi}_k = (\phi_1, \cdots, \phi_k)$ be the top k CCA vectors. Also denote $\widetilde{\mathbf{\Phi}}_k = \mathbf{\Phi}_k\mathbf{\Lambda}_k$ and $\widetilde{\mathbf{\Psi}}_k = \mathbf{\Psi}_k\mathbf{\Lambda}_k$. Then for any $k \times k$ orthogonal matrix $\mathbf{Q}$, $(\mathbf{\Phi}_k, \mathbf{\Psi}_k, \widetilde{\mathbf{\Phi}}_k, \widetilde{\mathbf{\Psi}}_k)\mathbf{Q}$ is a fixed point of AppGrad scheme.*

The top k dimensional canonical subspace is identifiable up to a rotation matrix and Proposition 2.3 shows that every optimum is a fixed point of *AppGrad* scheme.

2.4. Kernelization

Sometimes CCA is restricted because of its linearity and kernel CCA offers an alternative by projecting data into a high dimensional feature space. In this section, we show that *AppGrad* works for kernel CCA as well. Let $K_{\mathcal{X}}(\cdot,\cdot)$ and $K_{\mathcal{Y}}(\cdot,\cdot)$ be Mercer kernels, then there exists feature mappings $f_{\mathcal{X}} : \mathcal{X} \to \mathcal{F}_{\mathcal{X}}$ and $f_{\mathcal{Y}} : \mathcal{Y} \to \mathcal{F}_{\mathcal{Y}}$ such that $K_{\mathcal{X}}(x_i, x_j) = \langle f_{\mathcal{X}}(x_i), f_{\mathcal{X}}(x_j)\rangle$ and $K_{\mathcal{Y}}(y_i, y_j) = \langle f_{\mathcal{Y}}(y_i), f_{\mathcal{Y}}(y_j)\rangle$. Let $\mathbf{F}_{\mathcal{X}} = (f_{\mathcal{X}}(x_1), \cdots, f_{\mathcal{X}}(x_n))^\top$ and $\mathbf{F}_{\mathcal{Y}} = (f_{\mathcal{Y}}(y_1), \cdots, f_{\mathcal{Y}}(y_n))^\top$ be the compact representation of the objects in the possibly infinite dimensional feature space. Since the top k dimensional canonical vectors lie in the space spaned by the features, say $\mathbf{\Phi}_k = \mathbf{F}_{\mathcal{X}}^\top \mathbf{W}_{\mathcal{X}}$ and $\mathbf{\Psi}_k = \mathbf{F}_{\mathcal{Y}}^\top \mathbf{W}_{\mathcal{Y}}$ for some $\mathbf{W}_{\mathcal{X}}, \mathbf{W}_{\mathcal{Y}} \in \mathbb{R}^{n\times k}$. Let $\mathbf{K}_{\mathcal{X}} = \mathbf{F}_{\mathcal{X}}\mathbf{F}_{\mathcal{X}}^\top, \mathbf{K}_{\mathcal{Y}} = \mathbf{F}_{\mathcal{Y}}\mathbf{F}_{\mathcal{Y}}^\top$ be the Gram matrices. Similar to Lemma 2.1, kernel CCA can be formulated as

$$\arg\max_{\mathbf{W}_{\mathcal{X}}, \mathbf{W}_{\mathcal{Y}}} \quad \|\mathbf{K}_{\mathcal{X}}\mathbf{W}_{\mathcal{X}} - \mathbf{K}_{\mathcal{Y}}\mathbf{W}_{\mathcal{Y}}\|_F^2$$
$$\text{subject to } \mathbf{W}_{\mathcal{X}}^\top \mathbf{K}_{\mathcal{X}}\mathbf{K}_{\mathcal{X}}\mathbf{W}_{\mathcal{X}} = \mathbf{I}_k \quad \mathbf{W}_{\mathcal{Y}}^\top \mathbf{K}_{\mathcal{Y}}\mathbf{K}_{\mathcal{Y}}\mathbf{W}_{\mathcal{Y}} = \mathbf{I}_k$$

Following the same logic as Proposition 2.3, a similar fixed point result can be proved. Therefore, Algorithm 4 can be directly applied to compute $\mathbf{W}_{\mathcal{X}}, \mathbf{W}_{\mathcal{Y}}$ by simply replacing $\mathbf{X}, \mathbf{Y}$ with $\mathbf{K}_{\mathcal{X}}, \mathbf{K}_{\mathcal{Y}}$.

3. Stochastic *AppGrad*

Recently, there is a growing interest in stochastic optimization which is shown to have better performance for large-scale learning problems (Bousquet & Bottou, 2008; Bottou, 2010). Especially in the so-called 'data laden regime', where data is abundant and the bottleneck is runtime, stochastic optimization dominate batch algorithms

Algorithm 5 CCA via Stochastic *AppGrad* (Rank-k)

Input: Data matrix $\mathbf{X} \in \mathcal{R}^{n\times p_1}, \mathbf{Y} \in \mathcal{R}^{n\times p_2}$, initialization $(\mathbf{\Phi}^0, \mathbf{\Psi}^0, \widetilde{\mathbf{\Phi}}^0, \widetilde{\mathbf{\Psi}}^0)$, step size η_{1t}, η_{2t}, minibatch size m

Output : $(\mathbf{\Phi}_{\text{SAG}}, \mathbf{\Psi}_{\text{SAG}}, \widetilde{\mathbf{\Phi}}_{\text{SAG}}, \widetilde{\mathbf{\Psi}}_{\text{SAG}})$

repeat

Randomly pick a subset $\mathcal{I} \subset \{1, 2, \cdots, n\}$ of size m

$\widetilde{\mathbf{\Phi}}^{t+1} = \widetilde{\mathbf{\Phi}}^t - \eta_{1t}\mathbf{X}_{\mathcal{I}}^\top(\mathbf{X}_{\mathcal{I}}\widetilde{\mathbf{\Phi}}^t - \mathbf{Y}_{\mathcal{I}}\mathbf{\Psi}^t)/m$

SVD: $(\widetilde{\mathbf{\Phi}}^{t+1})^\top(\frac{1}{m}\mathbf{X}_{\mathcal{I}}^\top\mathbf{X}_{\mathcal{I}})\widetilde{\mathbf{\Phi}}^{t+1} = \mathbf{U}_x^\top\mathbf{D}_x\mathbf{U}_x$

$\mathbf{\Phi}^{t+1} = \widetilde{\mathbf{\Phi}}^{t+1}\mathbf{U}_x^\top\mathbf{D}_x^{-\frac{1}{2}}\mathbf{U}_x$

$\widetilde{\mathbf{\Psi}}^{t+1} = \widetilde{\mathbf{\Psi}}^t - \eta_{2t}\mathbf{Y}_{\mathcal{I}}^\top(\mathbf{Y}_{\mathcal{I}}\widetilde{\mathbf{\Psi}}^t - \mathbf{X}_{\mathcal{I}}\mathbf{\Phi}^t)/m$

SVD: $(\widetilde{\mathbf{\Psi}}^{t+1})^\top(\frac{1}{m}\mathbf{Y}_{\mathcal{I}}^\top\mathbf{Y}_{\mathcal{I}})\widetilde{\mathbf{\Psi}}^{t+1} = \mathbf{U}_y^\top\mathbf{D}_y\mathbf{U}_y$

$\mathbf{\Psi}^{t+1} = \widetilde{\mathbf{\Psi}}^{t+1}\mathbf{U}_y^\top\mathbf{D}_y^{-\frac{1}{2}}\mathbf{U}_y$

until convergence

both empirically and theoretically. Given these advantages, lots of efforts have been spent on developing stochastic algorithms for principal component analysis (Oja & Karhunen, 1985; Arora et al., 2012; Mitliagkas et al., 2013; Balsubramani et al., 2013). Despite promising progress in PCA, as mentioned in (Arora et al., 2012), stochastic CCA is more challenging and remains an open problem due to the whitening step.

As a gradient scheme, *AppGrad* naturally generalizes to the stochastic regime and we summarize in Algorithm 5. Compared with the batch version, only a small subset of samples are used to compute the gradient, which reduces the computational cost per iteration from $O(n(p_1 + p_2)k)$ to $O(m(p_1 + p_2)k)$ ($m = |\mathcal{I}|$ is the size of the minibatch). Empirically, this makes stochastic *AppGrad* much faster than the batch version as we will see in the experiments. Also, for large scale applications when fully calculating the CCA subspace is prohibitive, stochastic *AppGrad* can generate a decent approximation given a fixed computational power, while other algorithms only give a one-shot estimate after the whole procedure is carried out completely. Moreover, when there is a generative model, as shown in (Bousquet & Bottou, 2008), due to the tradeoff between statistical and numerical accuracy, fully solving an empirical risk minimization is unnecessary since the statistical error will finally dominate. On the contrary, stochastic optimization directly tackles the problem in the population level and therefore is more statistically efficient.

It is worth mentioning that the normalization step is accomplished using a sampled Gram matrix $\frac{1}{m}\mathbf{X}_{\mathcal{I}}^\top\mathbf{X}_{\mathcal{I}}$ and $\frac{1}{m}\mathbf{Y}_{\mathcal{I}}^\top\mathbf{Y}_{\mathcal{I}}$. A key observation is that when $m \in O(k)$, $(\widetilde{\mathbf{\Phi}}^{t+1})^\top(\frac{1}{m}\mathbf{X}_{\mathcal{I}}^\top\mathbf{X}_{\mathcal{I}})\widetilde{\mathbf{\Phi}}^{t+1} \approx (\widetilde{\mathbf{\Phi}}^{t+1})^\top(\frac{1}{m}\mathbf{X}^\top\mathbf{X})\widetilde{\mathbf{\Phi}}^{t+1}$ using standard concentration inequality, because the matrix we want to approximate $(\widetilde{\mathbf{\Phi}}^{t+1})^\top(\frac{1}{m}\mathbf{X}^\top\mathbf{X})\widetilde{\mathbf{\Phi}}^{t+1}$ is a $k\times k$ matrix, while generally $O(p)$ sample is needed to have $\frac{1}{m}\mathbf{X}_{\mathcal{I}}^\top\mathbf{X}_{\mathcal{I}} \approx \frac{1}{n}\mathbf{X}^\top\mathbf{X}$. As we have argued in previous section, this bonus is a byproduct of the fact that *AppGrad* tries to identify and whiten the directions that contains the CCA subspace simultaneously, or else $O(p)$ samples are necessary for whitening the whole data matrices.

4. Experiments

In this section, we present experiments on four real datasets to evaluate the effectiveness of the proposed algorithms for computing the top 20 (k=20) dimensional canonical subspace. A short summary of the datasets is in Table 1.

Mediamill is an annotated video dataset from the Mediamill Challenge (Snoek et al., 2006). Each image is a representative keyframe of a video shot annotated with 101 labels and consists of 120 features. CCA is performed to explore the correlation structure between the images and its labels.

MNIST is a database of handwritten digits. CCA is used to learn correlated representations between the left and right halves of the images.

Penn Tree Bank dataset is extracted from Wall Street Journal, which consists of 1.17 million tokens and a vocabulary size of $43,000$ (Lamar et al., 2010). CCA has been successfully used on this dataset to build low dimensional word embeddings (Dhillon et al., 2011; 2012). The task here is a CCA between words and their context. We only consider the $10,000$ most frequent words to avoid sample sparsity.

URL Reputation dataset (Ma et al., 2009) is extracted from UCI machine learning repository. The dataset contains 2.4 million URLs each represented by 3.2 million features. For simplicity we only use the first 2 million samples. 38% of the features are host based features like WHOIS info, IP prefix and 62% are lexical based features like Hostname and Primary domain. We run a CCA between a subset of host based features and a subset of lexical based features.

4.1. Implementations

Evaluation Criterion: The evaluation criterion we use for the first three datasets (Mediamill, MNIST, Penn Tree Bank) is *Proportions of Correlations Captured* (PCC). To introduce this term, we first define *Total Correlations Captured* (TCC) between two matrices to be the sum of their canonical correlations as defined in Lemma 1.1. Then, for estimated top k dimensional canonical subspace $\widehat{\mathbf{\Phi}}_k, \widehat{\mathbf{\Psi}}_k$ and true leading k dimensional CCA subspace $\mathbf{\Phi}_k, \mathbf{\Psi}_k$, PCC is defined as

$$\text{PCC} = \frac{\text{TCC}(\mathbf{X}\widehat{\mathbf{\Phi}}_k, \mathbf{Y}\widehat{\mathbf{\Psi}}_k)}{\text{TCC}(\mathbf{X}\mathbf{\Phi}_k, \mathbf{Y}\mathbf{\Psi}_k)}$$

Table 1. Brief Summary of Datasets

DATASETS	DESCRIPTION	p_1	p_2	n
MEDIAMILL	IMAGE AND ITS LABELS	100	120	$30,000$
MNIST	LEFT AND RIGHT HALVES OF IMAGES	392	392	$60,000$
PENN TREE BANK	WORD CO-OCURRANCE	$10,000$	$10,000$	$500,000$
URL REPUTATION	HOST AND LEXICAL BASED FEATURES	$100,000$	$100,000$	$1,000,000$

Intuitively PCC characterizes the proportion of correlations captured by certain algorithm compared with the true CCA subspace. Therefore, the higher is PCC the better is the estimated CCA subspace. This criterion has two major advantages over subspace distance $\|P_{\widehat{\mathbf{\Phi}}_k} - P_{\mathbf{\Phi}_k}\|$ (P_Ω is projection matrix of the column space of Ω). First, it is more natural and relevant considering that the goal of CCA is to capture most correlations between two data matrices. Second, when the eigengap $\Delta\lambda = \lambda_k - \lambda_{k+1}$ is not big enough, the top k dimensional CCA subspace is ill posed while the correlations captured is well defined.

We use the output of the standard spectral algorithms as the truth $(\mathbf{\Phi}_k, \mathbf{\Psi}_k)$ to calculate the denominator of PCC. However, for URL Reputation dataset, the number of samples and features are too large for the algorithm to compute the true CCA subspace in a reasonable amount of time and instead we only compare the numerator $\text{TCC}(\mathbf{X}\widehat{\mathbf{\Phi}}_k, \mathbf{Y}\widehat{\mathbf{\Psi}}_k)$ (monotone w.r.t. PCC) for different algorithms.

Initialization We initialize $(\mathbf{\Phi}^0, \mathbf{\Psi}^0)$ by first drawing $i.i.d.$ samples from standard Gaussian distribution and then normalize such that $(\mathbf{\Phi}^0)^\top \mathbf{S_x} \mathbf{\Phi}^0 = I_k$ and $(\mathbf{\Psi}^0)^\top \mathbf{S_y} \mathbf{\Psi}^0 = I_k$

Stepsize For both *AppGrad* and stochastic *AppGrad*, a small part of the training set is held out and cross-validation is used to choose the step size adaptively.

Regularization For all the algorithms, a little regularization is added for numerical stability which means we replace Gram matrix $\mathbf{X}^\top\mathbf{X}$ with $\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I}$ for some small positive λ.

Oversampling Oversampling means when aiming for top k dimensional subspace, people usually computes top $k+l$ dimesional subspace from which a best k diemsional subspace is extracted. In practice, $l = 5 \sim 10$ suffices to improve the performance. We only do a oversampling of 5 in the URL dataset.

4.2. Summary of Results

For the first three datasets (Mediamill, MNIST, Penn Tree Bank), both in-sample and out-of-sample PCC are computed for *AppGrad* and Stochastic *AppGrad* as summarized in Figure1. As you can see, both algorithms nearly capture most of the correlations compared with the true CCA subspace and stochastic *AppGrad* consistently achieves same PCC with much less computational cost than its batch version. Moreover, the larger is the size of the data, the bigger advantage will stochastic *AppGrad* obtain. One thing to notice is that, as revealed in Mediamill dataset, out-of-sample PCC is not necessarily less than in-sample PCC because both denominator and numerator will change on the hold out set.

For URL Reputation dataset, as we mentioned earlier, classical algorithms fails on a typical desktop. The reason is that these algorithms only produce a one-shot estimate after the whole procedure is completed, which is usually prohibitive for huge datasets. In this scenario, the advantage of online algorithms like stochastic *AppGrad* becomes crucial. Further, the stochastic nature makes the algorithm cost-effective and generate decent approximations given fixed computational resources (e.g. FLOP). As revealed by Figure 2, as the number of iterations increases, stochastic *AppGrad* captures more and more correlations.

Since the true CCA subspaces for URL dataset is too slow to compute, we compare our algorithm with some naive heuristics which can be carried out efficiently in large scale and catches a reasonable amount of correlation. Below is a brief description of them.

- Non-Whitening (NW-CCA): directly perform SVD on the unwhitened covariance matrix $\mathbf{X}^T\mathbf{Y}$. This strategy is also used in (Witten et al., 2009)
- Diagnoally Whitening (DW-CCA) (Lu & Foster, 2014): avoid inverting matrices by approximating $\mathbf{S_x}^{-\frac{1}{2}}, \mathbf{S_y}^{-\frac{1}{2}}$ with $(\text{diag}(\mathbf{S_x}))^{-\frac{1}{2}}$ and $(\text{diag}(\mathbf{S_y}))^{-\frac{1}{2}}$.
- Whitening the leading m Principal Component Directions (PCA-CCA): First compute the leading m dimensional principal component subspace and project the data matrices $\mathbf{X}$ and $\mathbf{Y}$ to the subspace, denote them $\mathbf{U}_x$ and $\mathbf{U}_y$. Then compute the top k dimensional CCA subspace of the pair $(\mathbf{U}_x, \mathbf{U}_y)$. At last, transform the CCA subspace of $(\mathbf{U}_x, \mathbf{U}_y)$ back to the CCA subspace of orginal matrix pair $(\mathbf{X}, \mathbf{Y})$. Specifically for this example, we choose $m = 1200$ (log(FLOP)=35, dominating the computational cost of Stochastic *AppGrad*) .

Remark 4.1. *For all the heuristics mentioned above, SVD*

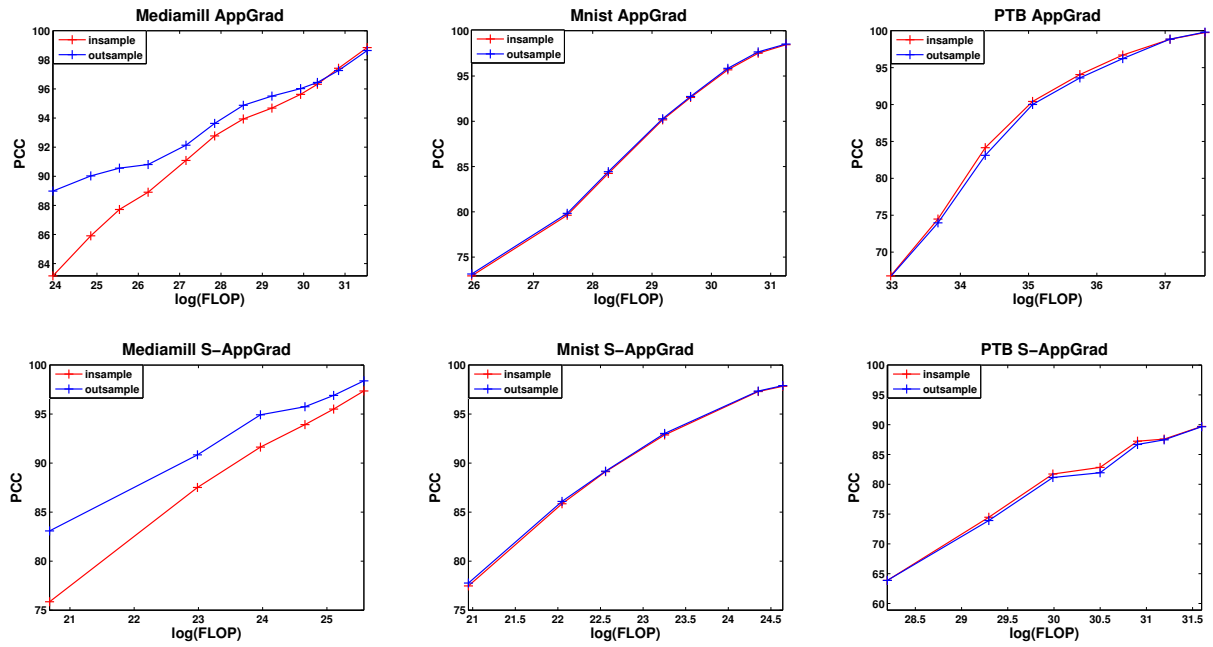

Figure 1. Proportion of Correlations Captured (PCC) by *AppGrad* and stochastic *AppGrad* on different datasets

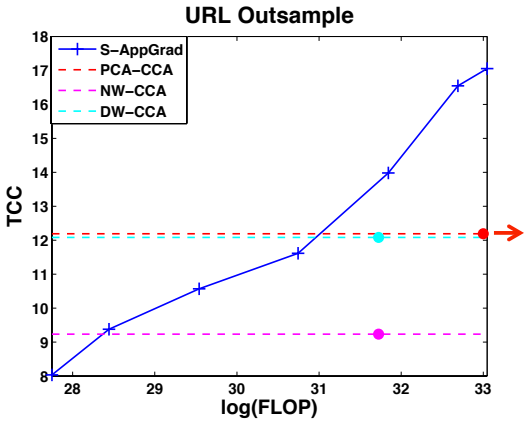

Figure 2. Total Correlations Captured (TCC) by NW-CCA, DW-CCA, PCA-CCA and stochastic *AppGrad* on URL dataset. The dash lines indicate TCC for those heuristics and the colored dots denote corresponding computational cost. Red arrow means log(FLOP) of PCA-CCA is more than 33.

and PCA steps are carried out using the randomized algorithms in (Halko et al., 2011). For PCA-CCA, as the number of Principal Components (m) increases, more correlation will be captured but the computational cost will also increase. When $m = p$, PCA-CCA is reduced to the orginal CCA.

Essentially, all the heuristics are incorrect algorithms and try to approximately whiten the data matrices. As suggested by Figure 2, stochastic *AppGrad* significantly captures much more correlations.

5. Conclusions and Future Work

In this paper, we present a novel first order method, *AppGrad*, to tackle large scale CCA as a nonconvex optimization problem. This bottleneck-free algorithm is both memory efficient and computationally scalable. More importantly, its online variant is well-suited to practical high dimensional applications where batch algorithm is prohibitive and data laden regime where data is abundant and runtime is main concern.

Further, *AppGrad* is flexible and structure information can be easily incorporated into the algorithm. For example, if the canonical vectors are assumed to be sparse (Witten et al., 2009; Gao et al., 2014), a thresholding step can be added between the gradient step and normalization step to obtain sparse solutions while it is hard to add sparse constraint to the classical CCA formulation which is a generalized eigenvalue problem. Heuristics in (Witten et al., 2009) avoid this by simply skipping the whitening procedure (NW-CCA). (Gao et al., 2014) resorts to semidefinite programming and therefore is very slow. *AppGrad* with thresholding works well in simulations and we leave its theoretical properties for future research.

References

Arora, R, Cotter, A, Livescu, K, and Srebro, N. Stochastic optimization for pca and pls. In *Communication, Control, and Computing (Allerton), 2012 50th Annual Allerton Conference on*, pp. 861–868. IEEE, 2012.

Avron, Haim, Boutsidis, Christos, Toledo, Sivan, and Zouzias, Anastasios. Efficient dimensionality reduction for canonical correlation analysis. In *ICML (1)*, pp. 347–355, 2013.

Balsubramani, Akshay, Dasgupta, Sanjoy, and Freund, Yoav. The fast convergence of incremental pca. In *Advances in Neural Information Processing Systems*, pp. 3174–3182, 2013.

Björck, Åke and Golub, Gene H. Numerical methods for computing angles between linear subspaces. *Mathematics of Computation*, pp. 579–594, 1973.

Blaschko, Matthew B and Lampert, Christoph H. Correlational spectral clustering. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pp. 1–8. IEEE, 2008.

Bottou, Léon. Large-Scale Machine Learning with Stochastic Gradient Descent. In Lechevallier, Yves and Saporta, Gilbert (eds.), *Proceedings of the 19th International Conference on Computational Statistics (COMPSTAT'2010)*, pp. 177–187, Paris, France, August 2010. Springer.

Bousquet, Olivier and Bottou, Léon. The tradeoffs of large scale learning. In *Advances in neural information processing systems*, pp. 161–168, 2008.

Chaudhuri, Kamalika, Kakade, Sham M, Livescu, Karen, and Sridharan, Karthik. Multi-view clustering via canonical correlation analysis. In *Proceedings of the 26th annual international conference on machine learning*, pp. 129–136. ACM, 2009.

Chen, Xi, Liu, Han, and Carbonell, Jaime G. Structured sparse canonical correlation analysis. In *International Conference on Artificial Intelligence and Statistics*, pp. 199–207, 2012.

Dhillon, Paramveer S., Foster, Dean, and Ungar, Lyle. Multi-view learning of word embeddings via cca. In *Advances in Neural Information Processing Systems (NIPS)*, volume 24, 2011.

Dhillon, Paramveer S., Rodu, Jordan, Foster, Dean P., and Ungar, Lyle H. Two step cca: A new spectral method for estimating vector models of words. In *Proceedings of the 29th International Conference on Machine learning*, ICML'12, 2012.

Foster, Dean P., Kakade, Sham M., and Zhang, Tong. Multi-view dimensionality reduction via canonical correlation analysis. Technical report, 2008.

Gao, Chao, Ma, Zongming, and Zhou, Harrison H. Sparse cca: Adaptive estimation and computational barriers. *arXiv preprint arXiv:1409.8565*, 2014.

Golub, Gene H and Zha, Hongyuan. *The canonical correlations of matrix pairs and their numerical computation*. Springer, 1995.

Halko, N., Martinsson, P. G., and Tropp, J. A. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Rev.*, 53(2):217–288, May 2011. ISSN 0036-1445.

Hariharan, Cibe and Subramanian, Shivashankar. Large scale multi-view learning on mapreduce. 2014.

Hotelling, H. Relations between two sets of variables. *Biometrika*, 28:312–377, 1936.

Kakade, Sham M. and Foster, Dean P. Multi-view regression via canonical correlation analysis. In *In Proc. of Conference on Learning Theory*, 2007.

Lamar, Michael, Maron, Yariv, Johnson, Mark, and Bienenstock, Elie. SVD and Clustering for Unsupervised POS Tagging. In *Proceedings of the ACL 2010 Conference Short Papers*, pp. 215–219, Uppsala, Sweden, 2010. Association for Computational Linguistics.

Liberty, Edo, Woolfe, Franco, Martinsson, Per-Gunnar, Rokhlin, Vladimir, and Tygert, Mark. Randomized algorithms for the low-rank approximation of matrices. *Proceedings of the National Academy of Sciences*, 104(51): 20167–20172, 2007.

Lu, Yichao and Foster, Dean P. Large scale canonical correlation analysis with iterative least squares. In *Advances in Neural Information Processing Systems*, pp. 91–99, 2014.

Ma, Justin, Saul, Lawrence K., Savage, Stefan, and Voelker, Geoffrey M. Identifying suspicious urls: An application of large-scale online learning. In *In Proc. of the International Conference on Machine Learning (ICML*, 2009.

McWilliams, Brian, Balduzzi, David, and Buhmann, Joachim. Correlated random features for fast semi-supervised learning. In *Advances in Neural Information Processing Systems*, pp. 440–448, 2013.

Mitliagkas, Ioannis, Caramanis, Constantine, and Jain, Prateek. Memory limited, streaming pca. In *Advances in Neural Information Processing Systems*, pp. 2886–2894, 2013.

Oja, Erkki and Karhunen, Juha. On stochastic approximation of the eigenvectors and eigenvalues of the expectation of a random matrix. *Journal of mathematical analysis and applications*, 106(1):69–84, 1985.

Sarlos, Tamas. Improved approximation algorithms for large matrices via random projections. In *Foundations of Computer Science, 2006. FOCS'06. 47th Annual IEEE Symposium on*, pp. 143–152. IEEE, 2006.

Shalev-Shwartz, Shai and Srebro, Nathan. Svm optimization: inverse dependence on training set size. In *Proceedings of the 25th international conference on Machine learning*, pp. 928–935. ACM, 2008.

Snoek, Cees GM, Worring, Marcel, Van Gemert, Jan C, Geusebroek, Jan-Mark, and Smeulders, Arnold WM. The challenge problem for automated detection of 101 semantic concepts in multimedia. In *Proceedings of the 14th annual ACM international conference on Multimedia*, pp. 421–430. ACM, 2006.

Witten, Daniela M, Tibshirani, Robert, and Hastie, Trevor. A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis. *Biostatistics*, pp. kxp008, 2009.

Woolfe, Franco, Liberty, Edo, Rokhlin, Vladimir, and Tygert, Mark. A fast randomized algorithm for the approximation of matrices. *Applied and Computational Harmonic Analysis*, 25(3):335–366, 2008.